Index Enhancements In SQL Server 2000

Lubor Kollar
SQL Server Query Processor PM
Microsoft Corporation

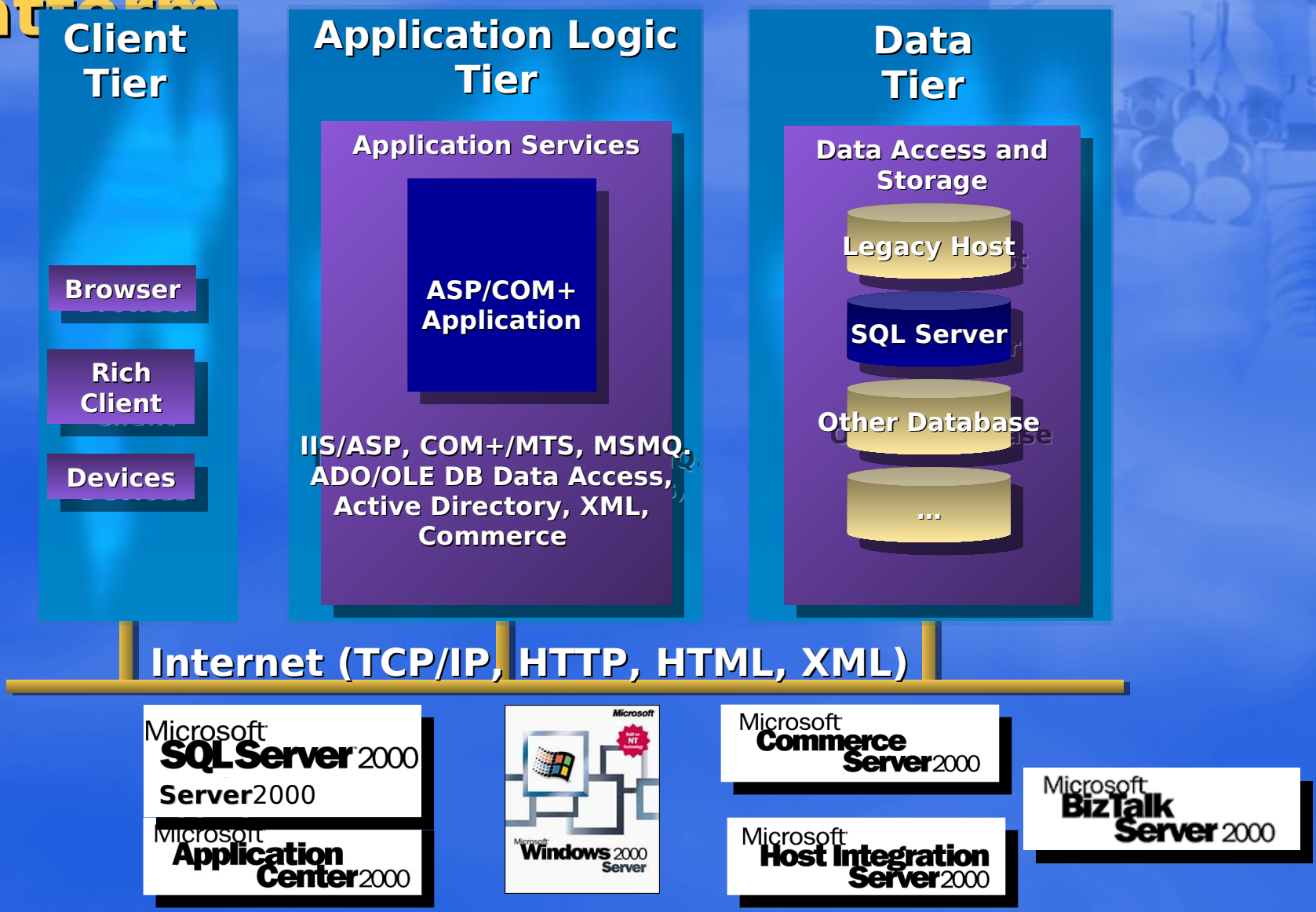

POWER UP
Windows DNA 2000
Readiness Conference
featuring SQL Server 2000

Index Enhancements
Agenda

- Where does it fit in the DNA context?
- Introduction to INDEXEs
- CREATE INEDEX processed as a Query
- Indexes on Computed Columns
- Indexes on Views
- Index Tuning Wizard
- Questions

Windows DNA 2000

Next Generation Web Application Platform



Index Enhancements
Introduction to INDEXES

- INDEXEs are created and maintained to gain query performance improvement
- Ideally users would like to see the performance improvement with minimal
  - Index creation and maintenance overhead
  - Cost of designing the “correct indexes”
  - Testing of various alternatives
- SQL Server 2000 is targeting

CREATE INDEX Processed As A Query

- CREATE INDEX special command in 7.0
- Integrated into the Query Processor in SQL Server 2000
  - Automatic parallelism with linear speedup
  - Optimization – may use existing index to reduce scan cost and sort
  - May specify ASC or DESC for each column

CREATE INDEX Processed As A Query

Examples

- Table T with existing index i1 on (a,b,c)
- CREATE INDEX i2 on T (b,c) will scan the index i1 instead of the whole table
- CREATE INDEX i3 on T (a,b) will scan the index i1 and will not perform any sort

CREATE INDEX Processed As A Query
Examples (ASC, DESC)

- SELECT a,b,c FROM t … ORDER BY a ASC, b DESC almost always requires a sort in 7.0
- SQL Server 2000 allows to create index to satisfy the ordering requirement thus eliminate the sort
  - CREATE INDEX i4 ON t (a ASC, b DESC, c ASC)

CREATE INDEX Processed As A Query

Best Practices for Big Tables

- Populate the table without any indexes
- Create the clustering index
- Create the non-clustering indexes
  - Both concurrent and sequential creation will work fine
  - If one index may take advantage of another index, create them in the appropriate order

Indexes On Computed Columns

- Computed Columns were introduced in 7.0
- SQL Server 2000 allows to create indexes on Computed Columns
  - The expression defining the Computed Column must be DETERMINISTIC
  - Certain SET options must be set
  - The above restrictions are explained
  - later in this presentation

Indexes On Computed Columns
Examples

- One year Sales Table S with SalesDate column
- SELECT SUM(SalesAmount) FROM S WHERE Datepart(mm,SalesDate)=12 requires always a table scan in 7.0

Indexes On Computed Columns
Examples

- ALTER TABLE S ADD SalesMonth as DatePart(mm,SalesDate)
- CREATE INDEX iSalesMonth on S (SalesMonth, SalesAmount)
- SELECT SUM(SalesAmount) FROM S WHERE SalesMonth=12 will use index seek in SQL Server 2000

Indexes on Views
Sub-agenda

- What is an index on a view
- What kind of scenarios, workloads and queries will benefit
- How to create an index on a view
- What are the conditions under which the index is considered for query execution
- Examples and Best Practice

Indexes On Views
What is an Index on a View

- A.K.A. Materialized Views
- The view may be a join, an aggregation or their combination
- Once the index is created the contents of the view is *persisted*
- The index is maintained automatically as any other index
- Optimizer may use the index

Indexes On Views
Benefiting workloads

- Decision support workloads
- Joins and aggregations of big tables
- Repeated patterns of queries
  - Repeated aggregations on the same or overlapping sets of column(s)
  - Repeated joins of the same tables on the same keys
  - Combinations of the above

Indexes On Views
Non-Benefiting workloads

- OLTP systems with many writes
- Databases with heavy updates
  - Updates are more expensive (similar to indexes, but may be worse)
- Expanding joins
  - Size of the indexed view may be too large
- Aggregations on high cardinality values
  - Size of the indexed view may be comparable with the original table

Indexes On Views
How to create Index on a View

- The first index on a view must be
  - CLUSTERED
  - UNIQUE
- CLUSTERED => the view is persisted like a table with clustered index
- UNIQUE => to ensure that we can maintain this index efficiently

Indexes On Views

How to create Index on a View - Considerations

- SET options consideration and DETERMINISM
- CREATE VIEW with SCHEMABINDING
- CREATE INDEX
- Optimizer considerations

Indexes On Views
DETERMINISM

- Some functions are *non-deterministic*, like getdate() – depends on the time it is executed
- If such function is persisted we may get different result if the same expression is executed or a stored value (say from yesterday) is used
- Therefore only DETERMINISTIC expressions may be used in *indexable* view definition

Indexes On Views
DETERMINISM

- Examples of DETERMINISTIC functions: SUM, AVG, +, -, /, *
- Examples of NONDETERMINISTIC functions: DATENAME (is language dependent), RAND()
- BOL and the notes on this slide have the complete list
- !!! *Implicit* string->date conversion is non-deterministic because it

Indexes On Views
SET options consideration and DETERMINISM

- But almost all expressions would be *non-deterministic* in the broad sense – for example
  - A/B AND A+B result depends on the current setting of ARITHABORT
  - String+NULL depends on the current setting of CONCAT_NULL_YIELDS_NULL
  - Etc…

Indexes On Views
SET options consideration

- The following set options MUST be fixed
- ARITHABORT, CONCAT_NULL_YIELDS_NULL, QUOTED_IDENTIFIER, ANSI_NULLS, ANSI_PADDING, ANSI_WARNING must be all ON
- NUMERIC_ROUNDABORT OFF

Indexes On Views
SET options consideration

- If the previously listed options are not set as required
  - The index on the view cannot be created
  - If the index is created and the options are changed at the time the query runs, the index will not be considered by the optimizer
  - Any maintenance (UPDATE, DELETE, INSERT) of a table

Indexes On Views
SET options Best Practices

- Avoid setting the 7 SET options in your application and sp's if you can and use the "default values"
- Use OLE DB or ODBC; then all your SET options are by default "*correct*" except of ARITHABORT
- Use sp_configure 'user options', 64
to set the ARITHABORT

Indexes On Views
Schemabinding

- Because the view is going to be persisted we must link it to all underlying tables
- Columns in the underlying tables cannot be dropped or modified (ALTER TABLE) if there is a view WITH SCHEMABINDING depending on them
- Only a view defined WITH SCHEMABINDING can be indexed

Indexes On Views
Schemabinding

- The fixed SET options listed before should be set appropriately already when the view is created (because some of them are recorded as the view property and the index would not be possible to create if they are not correct)
- To create a view with schemabinding, the user must have at least REFERENCES permission for all the tables in the view definition
- All tables and the view must be in

Indexes On Views
Create index

- The first index on the view with SCHEMABINDING must be
  - Clustered
  - Unique
- Only views without the following construct may be indexes: subqueries, outer joins, self join, DISTINCT, UNION, etc. (see the BOL for the complete list)
- The view creator must own all tables referenced in the view

Indexes On Views
Optimizer considerations

- User has limited control if an index on a view is used or not for a particular query (same for all indexes)
- If there are no hints, the index on a view is used if
  - The 7 SET options are set appropriately
  - The optimizer discovered a match between a sub-graph of the query graph and the view definition
  - Costing favors use of the index

Indexes On Views
Optimizer considerations - Hints

- If view is referenced in the query then the index may be enforced similarly as for a table
  - SELECT … FROM V1 WITH (NOEXPAND, INDEX(I1)) WHERE ….
  - NOEXPAND indicates not to expand the view definition
  - Index View usage cannot be enforced in queries where the View is not directly referenced
- OPTION (EXPAND VIEWS) at the end of the query disallows use of indexed views anywhere in the query plan

Indexes On Views

Example (against NORTHWIND db)

- More examples in the slide notes
- Identify 5 products with overall biggest discount total

```
select TOP 5 ProductID,
SUM(UnitPrice*Quantity*Discount) Rebate,
from [order details]
group by ProductID
order by Rebate desc
```

Indexes On Views

Example (against NORTHWIND db)

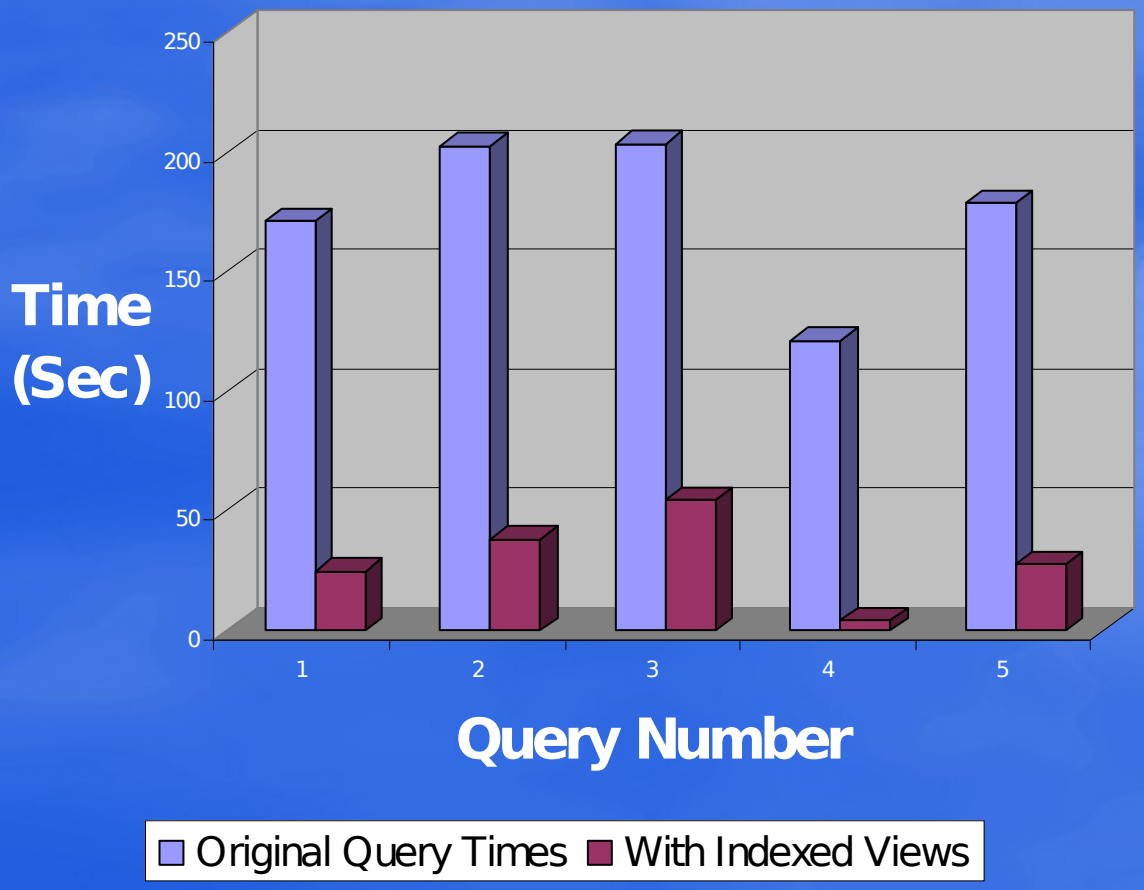
- Identify 5 products with overall biggest discount total

```
create view Vdiscount2 with
    schemabinding as
select SUM(UnitPrice*Quantity*Discount)
    SumDiscoutPrice2, COUNT_BIG(*) Count,
    ProductID
from dbo.[order details]
group By ProductID

create unique clustered index
    VDiscountInd on Vdiscount2 (ProductID)
```

Indexes On Views

Example (speedup on production database)



Indexes On Views
Optimizer considerations – Best Practices

- I have a query; how will I find if the Indexed View is used? What if not?
  - The query plan contains reference to the index on the view if the view is used
  - If the index on the view is not shown in the query plan, investigate
    - The cost of the query plan. If it is very small (especially if it is less than 1), we will not consider using Indexed Views
    - Setting of the mandatory 7 SET options in your session is not correct. Use SESSIONPROPERTY to investigate
    - Match between the View query graph and the Query query graph could not be established. Investigate the plan of the query and plan of SELECT * FROM *<your Indexed view>*

Indexes On Views
UDFs, Indexes on Computed Columns (ICCs)

- UDFs, ICCs and IVs have similar determinism considerations
- Only deterministic UDFs created WITH SCHEMABINDING may be indexed
- But deterministic UDFs cannot reference any tables, views, etc

Index Tuning Wizard
New Features

- Designing Indexed Views – optional selection
- Super-fast mode
- Sampling the dataset (this is now a new default)
- Command line interface (alongside the UI)

Conclusion

New SQL Server 2000 Index Features

- Parallel and “smart” index creation
- Indexes on Views and Computed Columns
- Enhanced Index Tuning Wizard

- ***Questions now? Ask, please***
- ***Questions later? E-mail to LuberK@microsoft.com***

POWER UP

Microsoft